# Software required

Download and install openssl from <https://wiki.openssl.org/index.php/Binaries>

Note: the one that comes with mingw / git-hub works if you run it from cmd, assuming when you installed it, you made those commands available to the cmd prompt.

Add it to PATH

Download openssl.cnf from <https://github.com/openssl/openssl/blob/master/apps/openssl.cnf>

Into openssl install directory (missing for some reason)

### CMD shell fixup

The instructions were tested in a Windows CMD shell.

To avoid this warning:

WARNING: can't open config file: /usr/local/ssl/openssl.cnf

you’ll want to set an environment variable to point to the openssl.cnf you just downloaded. Something like:

set OPENSSL\_CONF=c:\openssl-1.0.2k-i386-win32\openssl.cnf

This can be done either after opening each shell, or by adding an environment variable.

<WIN> edit sys <ENTER>

is usually enough to find the right dialog.

Doing this means you also don’t have to include the -config parameter on several of the openssl commands (like openssl req)

# Generate a self-signed cert with an RSA key

<http://stackoverflow.com/questions/10175812/how-to-create-a-self-signed-certificate-with-openssl#10176685>

<https://www.tbs-certificates.co.uk/FAQ/en/openssl-windows.html>

From a cmd shell:

openssl req -x509 -nodes -sha256 -newkey rsa:2048 -days 365 -keyout key.pem -out ddicert.pem -subj "/C=US/ST=Texas/L=College Station/O=Capsher Technology, Inc./OU=Sonic Team/CN=Sonic Digital-Drive-In/emailAddress=gprice@capsher.com"

### Check that the cert does all it should

It should answer Yes to all the purposes except timeserver

<https://stackoverflow.com/questions/21297139/how-do-you-sign-certificate-signing-request-with-your-certification-authority>

openssl x509 -purpose -inform PEM -in ddicert.pem

The results should be:

Certificate purposes:

SSL client : Yes

SSL client CA : Yes

SSL server : Yes

SSL server CA : Yes

Netscape SSL server : Yes

Netscape SSL server CA : Yes

S/MIME signing : Yes

S/MIME signing CA : Yes

S/MIME encryption : Yes

S/MIME encryption CA : Yes

CRL signing : Yes

CRL signing CA : Yes

Any Purpose : Yes

Any Purpose CA : Yes

OCSP helper : Yes

OCSP helper CA : Yes

Time Stamp signing : No

Time Stamp signing CA : Yes

And you should see a valid cert with at least 3 extensions when you run:

keytool -printcert -v -file ddicert.pem

### {Optional} Revert Private key format

The key is generated in the new style (“BEGIN PRIVATE KEY”, PKCS#10 in the openssl I’m using). The key we found in PEDS was using an older style (“BEGIN RSA PRIVATE KEY”, PKCS#1) <http://stackoverflow.com/questions/17733536/how-to-convert-a-private-key-to-an-rsa-private-key>

To convert to the older style (seems to work fine without doing this, but don’t want to lose the instructions just in case):

openssl rsa -in key.pem -out key.pem

### Create digital\_drive\_in.pem

Next concatenate the key and cert files together into a file called digital\_drive\_in.pem

Copy ddicert.pem + key.pem digital\_drive\_in.pem

Note: this is the opposite order of the original .pem; however, this still works, and allows easier import into java. Java’s keytool doesn’t seem to search the .pem for the certificate, it stops in the first section.

## Install New Cert/Key into PEDS

Back-up old and copy new pem

### First, replace PEDS’s Cert/Key file

set dest=c:\Sonic\bin\peds\webfiles

move %dest%\digital\_drive\_in.pem "%dest%\ddi.pem.%date:/=-%%time::=-%"

Copy digital\_drive\_in.pem %dest%\digital\_drive\_in.pem

### Second, replace DDI’s Cert/Key file

Set version=5.6.5.8

set dest=c:\Sonic\bin\digital\_drive\_in\versions\%version%\webfiles

move %dest%\digital\_drive\_in.pem "%dest%\ddi.pem.%date:/=-%%time::=-%"

Copy digital\_drive\_in.pem %dest%\digital\_drive\_in.pem

### Third, start up PEDS and make sure /current/traces are coming through to DDI

* **Make sure** C:\Sonic\conf\digital\_drive\_in\tuning-config.json
* And C:\Sonic\bin\peds\app-config.json
* Both point to port **8025** for the Web requests

Then start PEDS and look in DDI console for /current/traces requests (about every 5 seconds by default)

Start /wait c:\sonic\StartPOSandPOPS.bat

### Fourth, stop PEDS

Starting pos and pops should have opened up another cmd window in c:\sonic. In that one, run

stop<tab>

# 

# 

# Use digital\_drive\_in.pem with Java

Open cmd in admin mode:

<Win> cmd <CTRL>+<SHIFT>+<ENTER>

## Check Java

Make sure JAVA\_HOME is set to the first java.exe dir

Echo %JAVA\_HOME%

Where java

Make sure it’s the 32-bit java.

Java -d64 -version

You should get the following Error

Error: This Java instance does not support a 64-bit JVM.

Make sure to also set the IDE’s JRE to point to the same version as where java (if you run from one)

### If it’s a 64-bit java...

Change PATH to put the 32-bit java first (downloading if necessary).

<WIN> edit sys <ENTER>

is usually enough to find the right dialog.

Close the current shell and start over from a new one.

Make sure the IDE is using the same java.

## Prepare shell

If you didn’t set the Environment Variable globally (see top of doc):

set OPENSSL\_CONF=c:\openssl-1.0.2k-i386-win32\openssl.cnf

If JAVA\_HOME points to a jdk:

Pushd "%JAVA\_HOME%\jre\lib\security"

If JAVA\_HOME points to a JRE:

Pushd "%JAVA\_HOME%\lib\security"

If necessary, grab a copy of digital\_drive\_in.pem to play with:

Copy c:\sonic\bin\peds\webfiles\digital\_drive\_in.pem .

### Check file

Make sure the .pem file has the certificate first, followed by the RSA Private Key

Type digital\_drive\_in.pem

If not, use an editor to change the order

### Make a backup copy of cacerts

If JAVA\_HOME points to a jdk:

Copy "%JAVA\_HOME%\jre\lib\security"\cacerts "%JAVA\_HOME%\jre\lib\security\cacerts.orig.%date:/=-%%time::=-%"

If JAVA\_HOME points to a JRE:

Copy "%JAVA\_HOME%\lib\security"\cacerts "%JAVA\_HOME%\lib\security\cacerts.orig.%date:/=-%%time::=-%"

## Install digital\_drive\_in.pem into cacerts

3 options for letting java know about the ddi cert:

1. Create a custom truststore manager in code and suck the bytes in.
2. Put it by itself in jssecacerts, which is read before cacerts
3. Put it in with all the other “trusted ca” certs into cacerts

We’re picking Option 3 since otherwise IDEs fail and on the presumption that non-test environments will use a real signed cert, so we’ll need all those CA certs.

### Install cert

keytool -importcert -alias digital\_drive\_in -keystore cacerts -storepass changeit -noprompt -file digital\_drive\_in.pem

(see default truststore in <http://docs.oracle.com/javase/6/docs/technotes/guides/security/jsse/JSSERefGuide.html#InstallationAndCustomization>)

In spite of the param name (-keystore), jssecacerts is a \*truststore\*. It only holds certificates. When digital\_drive\_in.pem is imported, the private key is lost.

### Verify cert in truststore

keytool -list -v -keystore cacerts -storepass changeit | findstr /i “CAPSHER”

Note: This is one place where more exactitude can be warranted. If things fail, come back here and check that it’s the right (and only) capsher cert in this file!

## Convert .PEM to .P12 so Java can use private key

# Create PKCS12 keystore from private key and public certificate.

openssl pkcs12 -export -name digital\_drive\_in -in digital\_drive\_in.pem -inkey digital\_drive\_in.pem -out ddi.p12

* Export password changeit

## In java, need to set keystore manager to find ddi.p12.

Put it somewhere in the class path. E.g., if the java code is in an intellij project “forwarder

Copy ddi.p12 %USERPROFILE%\IdeaProjects\forwarder\src\main\resources\ddi.p12

And then add to main.java

String certFile = Main.class.getClassLoader().getResource("ddi.p12").getFile();

System.setProperty("javax.net.ssl.keyStore", certFile);

System.*setProperty*("javax.net.ssl.keyStorePassword", "changeit");

### Prepare Sonic

* Make sureC:\Sonic\conf\digital\_drive\_in\tuning-config.json
  + Web requests at port **9**025
* And C:\Sonic\bin\peds\app-config.json
  + Point to port **8**025for the Web requests
* That way, the dispatcher can start a server socket on port 8025 and forward to the ddi socket on 9025.
* Execute

Del /f c:\sonic\logs

### Change IntelliJ VM to use the one that runs PEDS (where java):

Run...Edit Configurations… JRE

C:\Program Files\Java\jdk1.8.0\_121

(or where ever which java tells you)

-- can also add VM Options (recommended if it doesn’t work : )

-Djavax.net.debug=ssl,handshake

When it works, the ddi log will have entries for /current/traces from the recent run.

Make sure truststore, keystore are right and cert is right. For problems, look for for “fatal” in the console when it runs and fails.

## Password

Always use

changeit

--

# 

# 

# Notes

see http://www.ateam-oracle.com/tls-and-java/

<https://www.stunnel.org/index.html>

-- for the DDI code example, (look up sslv23\_create whatever. It’s more than sslv2.3).

See pops-platform/cardfree\_sim/AppManager.cpp

And look for “cipher”.

In java…

http://stackoverflow.com/questions/18787419/ssl-socket-connection

Other links

<https://www.codeproject.com/Tips/1043003/Create-a-Simple-Web-Server-in-Java-HTTPS-Server>

<https://www.sslshopper.com/article-most-common-openssl-commands.html>

Wish i found this sooner. Defaults:

<http://docs.oracle.com/javase/6/docs/technotes/guides/security/jsse/JSSERefGuide.html#InstallationAndCustomization>

# Reverse .p12 back to .pem

Extract out of jssecacerts back into to [PKCS #12](https://en.wikipedia.org/wiki/PKCS_12), then to .pem

keytool -importkeystore -srckeystore jssecacerts -srcstorepass changeit -destkeystore out.p12 -deststoretype PKCS12 -srcalias digital\_drive\_in -deststorepass changeit -destkeypass changeit

Export certificate using openssl:

openssl pkcs12 -in out.p12 -nokeys -out outcert.pem

Export unencrypted private key:

openssl pkcs12 -in out.p12 -nodes -nocerts -out outkey.pem

## Delete entry from truststore (if didn’t start with empty one)

keytool -delete -alias digital\_drive\_in -keystore truststore -storepass changeit